

# TypeScript Meetup v0.2

10 min	Around the room introductions
10 min	Opening comments - Anders Hejlsberg
20 min	TypeScript in the trenches - Eric Gamma
20 min	TypeScript update - Luke Hoban
20 min	TypeScript Q&A session - Luke, Anders, Steve Lucco, and Team
20 min	<b>Break: Eat, drink, and Socialize</b>
10 min	Bing and TypeScript - Kiran Badam & Sarvesh Nagpal
10 min	Cloud Make: Using TypeScript for Builds and WorkFlows - Wolfram Schulte
10 min	TypeScript language experiments - Ron Buckton
10 min	Xbox Music and TypeScript - Dan Rodgers (not recorded/streamed)
10 min	Using TypeScript to simplify JavaScript hosting - Paul Vick (not recorded/streamed)

Contact: [stevenic](#)

MICROSOFT CONFIDENTIAL

Thank You!

# TypeScript Momentum

## Community

Over 3000 CodePlex posts, 100 forks, 1000 StackOverflow questions, and 300 feature requests

## Ecosystem

Over 180 .d.ts library definitions covering more than 90% of popular JavaScript frameworks.

## Tool Support

IDEs: WebStorm, WebMatrix, JSBin, Web Essentials, Sublime Text, vi, Emacs, FlashDevelop, Brackets

Build: Heroku, Ruby, grunt, ASP.NET, node, compile-in-client

Testing: Chutzpah, tsUnit

Preprocessing: Bakehouse

Typings management: tsd

# TypeScript 0.9

## Generics

Generic classes, interfaces, and methods: The top requested feature by users

## Overloading on Constants

Examples include `createElement`, `getElementsByTagName`, `addEventListener`

## Other Features

Enum types, 'export =', function/module and class/module merging, methods in object literals

## Compiler

Incremental parser, pull-model type checker, scaling to 100K+ line projects

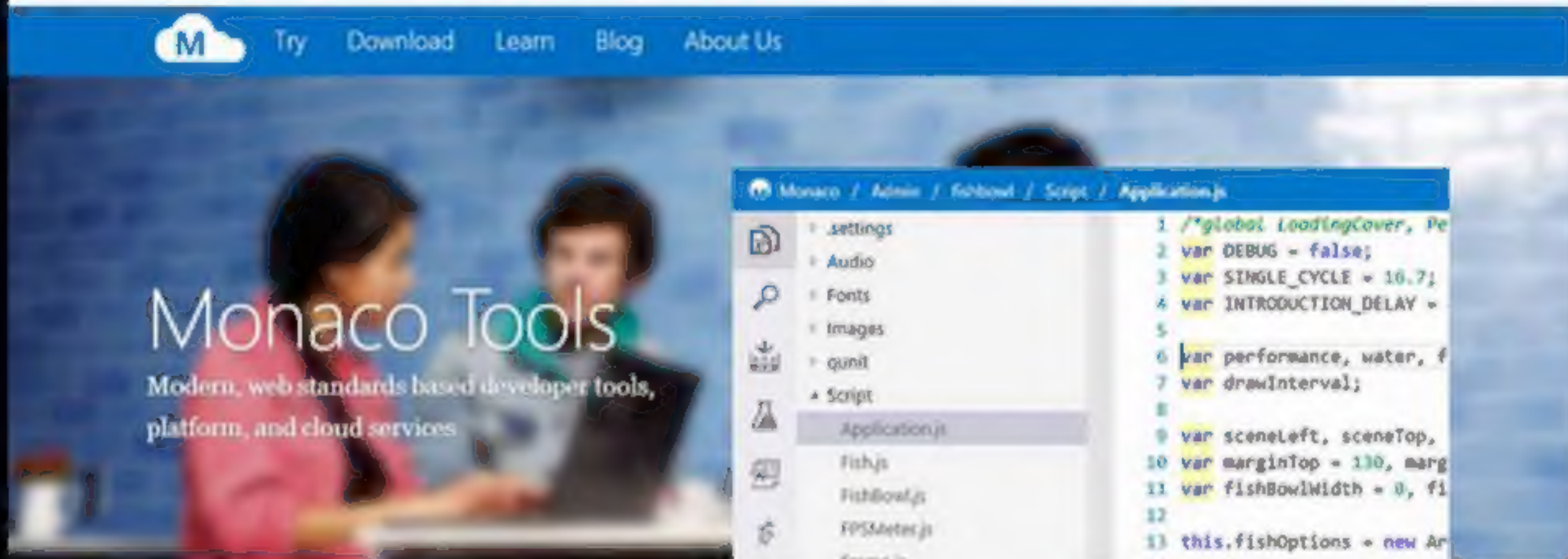


# Application Scale JavaScript

## TypeScript in the Trenches

Erich Gamma  
Microsoft

# http://monacotools



## Easily get started

No long waits for installation and configuration.  
Just log in and get coding!

## Code in the cloud with confidence

Develop online with rich, contextual features like IntelliSense, Source Assist, Quick Open and many more!

## Platform of reusable components

We've built Monaco as a set of simple, yet extensible modules that you can adopt and tweak yourself, based on your requirements and scenarios



# Team Foundation Server

The screenshot shows the Visual Studio interface with a diff comparison of the file `gotoSymbolHandler.js`. The left sidebar shows the file explorer with the project structure. The main editor area displays the diff between two versions of the file: `gotoSymbolHandler.js (ChangeSet 25253)` and `gotoSymbolHandler.js (ChangeSet 25254)`. The right pane shows the code for the newer version, `gotoSymbolHandler.js (ChangeSet 25254)`.


The code is in JavaScript and includes comments in German. The diff highlights changes in the `toQuickOpenEntries` function, specifically the `flatten` and `convertToEntries` steps.

The code in the right pane is as follows:

```

114     return vscode.supportsLanguageFeature(vscode.LanguageFeatureKind.SnippetSupport) ? gotoSymbolHandler.prototype.toQuickOpenEntries : gotoSymbolHandler.prototype.toQuickOpenEntries;
115 }
116
117     return false;
118 }
119
120     gotoSymbolHandler.prototype.autoSelectFirstEntry = function (searchValue) {
121         return searchValue.length > 0;
122     }
123
124     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
125         var results = [];
126         var searchValueLower = searchValue.toLowerCase();
127         // Flatten
128         var flattened = [];
129         if (outline) {
130             this.flatten(outline, flattened);
131         }
132         // Convert to Entries
133         for (var i = 0; i < flattened.length; i++) {
134             var element = flattened[i];
135             var highlights = this.getHighlights(element, searchValueLower);
136             if (highlights) {
137                 // Generate functions for label and description
138                 var label = element.label;
139                 if (element.type === 'method') {
140                     label = Strings.format(element.displayName, element.parameters);
141                 }
142                 // Show parent scope as well
143                 var description = null;
144                 if (element.parent) {
145                     description = element.parent.label;
146                 }
147                 results.push({
148                     label: label,
149                     description: description,
150                     uri: element.uri,
151                     range: element.range,
152                     kind: element.kind,
153                     highlights: highlights
154                 });
155             }
156         }
157         return results;
158     }
159
160     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
161         return this.toQuickOpenEntries(outline, searchValue);
162     }
163
164     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
165         return this.toQuickOpenEntries(outline, searchValue);
166     }
167
168     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
169         return this.toQuickOpenEntries(outline, searchValue);
170     }
171
172     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
173         return this.toQuickOpenEntries(outline, searchValue);
174     }
175
176     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
177         return this.toQuickOpenEntries(outline, searchValue);
178     }
179
180     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
181         return this.toQuickOpenEntries(outline, searchValue);
182     }
183
184     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
185         return this.toQuickOpenEntries(outline, searchValue);
186     }
187
188     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
189         return this.toQuickOpenEntries(outline, searchValue);
190     }
191
192     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
193         return this.toQuickOpenEntries(outline, searchValue);
194     }
195
196     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
197         return this.toQuickOpenEntries(outline, searchValue);
198     }
199
200     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
201         return this.toQuickOpenEntries(outline, searchValue);
202     }
203
204     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
205         return this.toQuickOpenEntries(outline, searchValue);
206     }
207
208     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
209         return this.toQuickOpenEntries(outline, searchValue);
210     }
211
212     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
213         return this.toQuickOpenEntries(outline, searchValue);
214     }
215
216     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
217         return this.toQuickOpenEntries(outline, searchValue);
218     }
219
220     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
221         return this.toQuickOpenEntries(outline, searchValue);
222     }
223
224     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
225         return this.toQuickOpenEntries(outline, searchValue);
226     }
227
228     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
229         return this.toQuickOpenEntries(outline, searchValue);
230     }
231
232     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
233         return this.toQuickOpenEntries(outline, searchValue);
234     }
235
236     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
237         return this.toQuickOpenEntries(outline, searchValue);
238     }
239
240     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
241         return this.toQuickOpenEntries(outline, searchValue);
242     }
243
244     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
245         return this.toQuickOpenEntries(outline, searchValue);
246     }
247
248     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
249         return this.toQuickOpenEntries(outline, searchValue);
250     }
251
252     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
253         return this.toQuickOpenEntries(outline, searchValue);
254     }
255
256     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
257         return this.toQuickOpenEntries(outline, searchValue);
258     }
259
260     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
261         return this.toQuickOpenEntries(outline, searchValue);
262     }
263
264     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
265         return this.toQuickOpenEntries(outline, searchValue);
266     }
267
268     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
269         return this.toQuickOpenEntries(outline, searchValue);
270     }
271
272     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
273         return this.toQuickOpenEntries(outline, searchValue);
274     }
275
276     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
277         return this.toQuickOpenEntries(outline, searchValue);
278     }
279
280     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
281         return this.toQuickOpenEntries(outline, searchValue);
282     }
283
284     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
285         return this.toQuickOpenEntries(outline, searchValue);
286     }
287
288     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
289         return this.toQuickOpenEntries(outline, searchValue);
290     }
291
292     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
293         return this.toQuickOpenEntries(outline, searchValue);
294     }
295
296     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
297         return this.toQuickOpenEntries(outline, searchValue);
298     }
299
300     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
301         return this.toQuickOpenEntries(outline, searchValue);
302     }
303
304     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
305         return this.toQuickOpenEntries(outline, searchValue);
306     }
307
308     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
309         return this.toQuickOpenEntries(outline, searchValue);
310     }
311
312     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
313         return this.toQuickOpenEntries(outline, searchValue);
314     }
315
316     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
317         return this.toQuickOpenEntries(outline, searchValue);
318     }
319
320     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
321         return this.toQuickOpenEntries(outline, searchValue);
322     }
323
324     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
325         return this.toQuickOpenEntries(outline, searchValue);
326     }
327
328     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
329         return this.toQuickOpenEntries(outline, searchValue);
330     }
331
332     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
333         return this.toQuickOpenEntries(outline, searchValue);
334     }
335
336     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
337         return this.toQuickOpenEntries(outline, searchValue);
338     }
339
340     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
341         return this.toQuickOpenEntries(outline, searchValue);
342     }
343
344     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
345         return this.toQuickOpenEntries(outline, searchValue);
346     }
347
348     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
349         return this.toQuickOpenEntries(outline, searchValue);
350     }
351
352     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
353         return this.toQuickOpenEntries(outline, searchValue);
354     }
355
356     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
357         return this.toQuickOpenEntries(outline, searchValue);
358     }
359
360     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
361         return this.toQuickOpenEntries(outline, searchValue);
362     }
363
364     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
365         return this.toQuickOpenEntries(outline, searchValue);
366     }
367
368     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
369         return this.toQuickOpenEntries(outline, searchValue);
370     }
371
372     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
373         return this.toQuickOpenEntries(outline, searchValue);
374     }
375
376     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
377         return this.toQuickOpenEntries(outline, searchValue);
378     }
379
380     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
381         return this.toQuickOpenEntries(outline, searchValue);
382     }
383
384     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
385         return this.toQuickOpenEntries(outline, searchValue);
386     }
387
388     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
389         return this.toQuickOpenEntries(outline, searchValue);
390     }
391
392     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
393         return this.toQuickOpenEntries(outline, searchValue);
394     }
395
396     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
397         return this.toQuickOpenEntries(outline, searchValue);
398     }
399
400     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
401         return this.toQuickOpenEntries(outline, searchValue);
402     }
403
404     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
405         return this.toQuickOpenEntries(outline, searchValue);
406     }
407
408     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
409         return this.toQuickOpenEntries(outline, searchValue);
410     }
411
412     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
413         return this.toQuickOpenEntries(outline, searchValue);
414     }
415
416     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
417         return this.toQuickOpenEntries(outline, searchValue);
418     }
419
420     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
421         return this.toQuickOpenEntries(outline, searchValue);
422     }
423
424     gotoSymbolHandler.prototype.toQuickOpenEntries = function (outline, searchValue) {
425         return this.toQuickOpenEntries(outline, searchValue);
426     }
427
428    
```

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
```

 **Good things**  
Martin Woodward - less than a month ago - [reply](#)

```

7 public static PalindromeJC palindrome
8 {
9     class Program
10    {
11        static void Main(string[] args)

```

**Line level (partner)**  
 Martin Woodward - less than a month ago - saving

```
10 Console.WriteLine("Full Name");
```

# dafny Microsoft Research

Is this program correct? Ask dafny!

```
1 function Ackermann(m: int, n: int): int
2   // The following lexicographic pair allows Dafny to prove termination.
3   // Still, you may not want to sit around and wait for a call to Ackermann
4   // to terminate.
5   decreases m, n;
6 {
7   if m <= 0 then
8     n + 1
9   else if n <= 0 then
10    Ackermann(m - 1, 1)
11  else
12    Ackermann(m - 1, Ackermann(m, n - 1))
13 }
14
```

ask dafny

home

tutorial

video

performance

Dafny program verifier finished with 1 verified, 0 errors

**samples**

Hello

Fibonacci

CountToN

**about Dafny - A language and program verifier for functional correctness**

Dafny is an imperative, object-oriented programming language with classes and inductive datatypes, and specification constructs for describing intended behavior. The Dafny verifier checks that programs live up to their specifications.



# TypeScript Playground

TypeScript

learn

play

get it

run it

join in

TypeScript

Greeter

Run

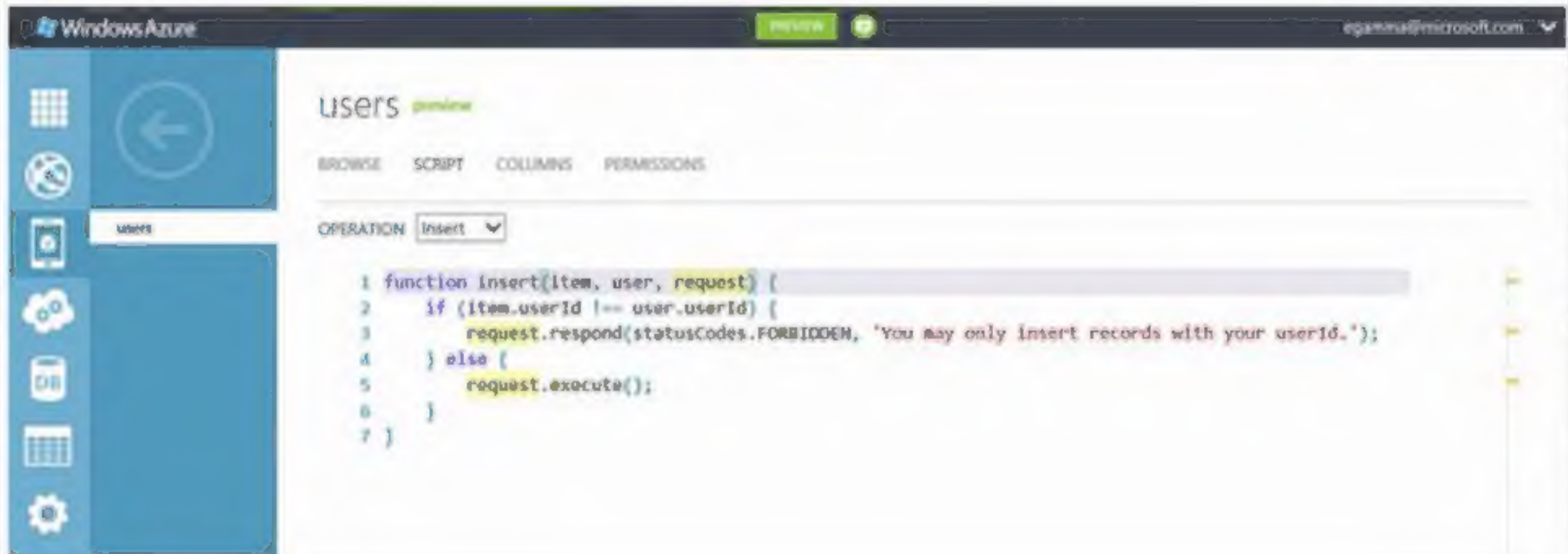
JavaScript

```
1 class Greeter {  
2     greeting: string;  
3  
4     constructor(greeting: string) {  
5         this.greeting = greeting;  
6     }  
7  
8     greet() {  
9         return "<div>" + this.greeting + "</div>";  
10    }  
11 }  
12  
13  
14  
15 var greeter = new Greeter("hi");  
16 var str = greeter.greet();  
17 document.body.innerHTML = str;
```

@greet () => string  
() => string  
@greeting

```
1 var Greeter = (function () {  
2     function Greeter(greeting) {  
3         this.greeting = greeting;  
4     }  
5     Greeter.prototype.greet = function () {  
6         return "<div>" + this.greeting + "</div>";  
7     };  
8     return Greeter;  
9 })();  
10 var greeter = new Greeter("hello, world!");  
11 var str = greeter.greet();  
12 document.body.innerHTML = str;  
13
```

# Azure



Windows Azure

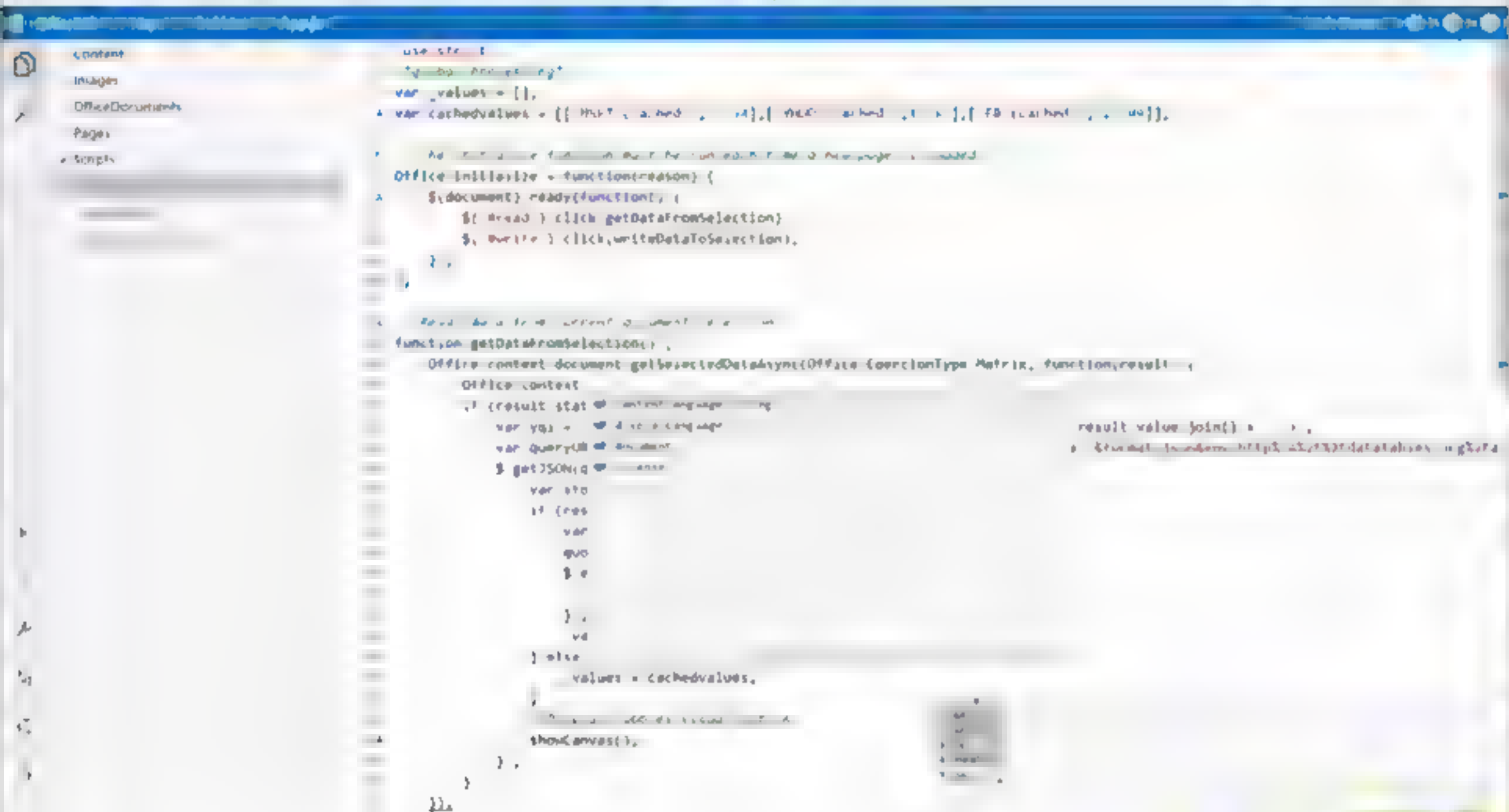
users [preview](#)

BROWSE SCRIPT COLLUMNS PERMISSIONS

OPERATION

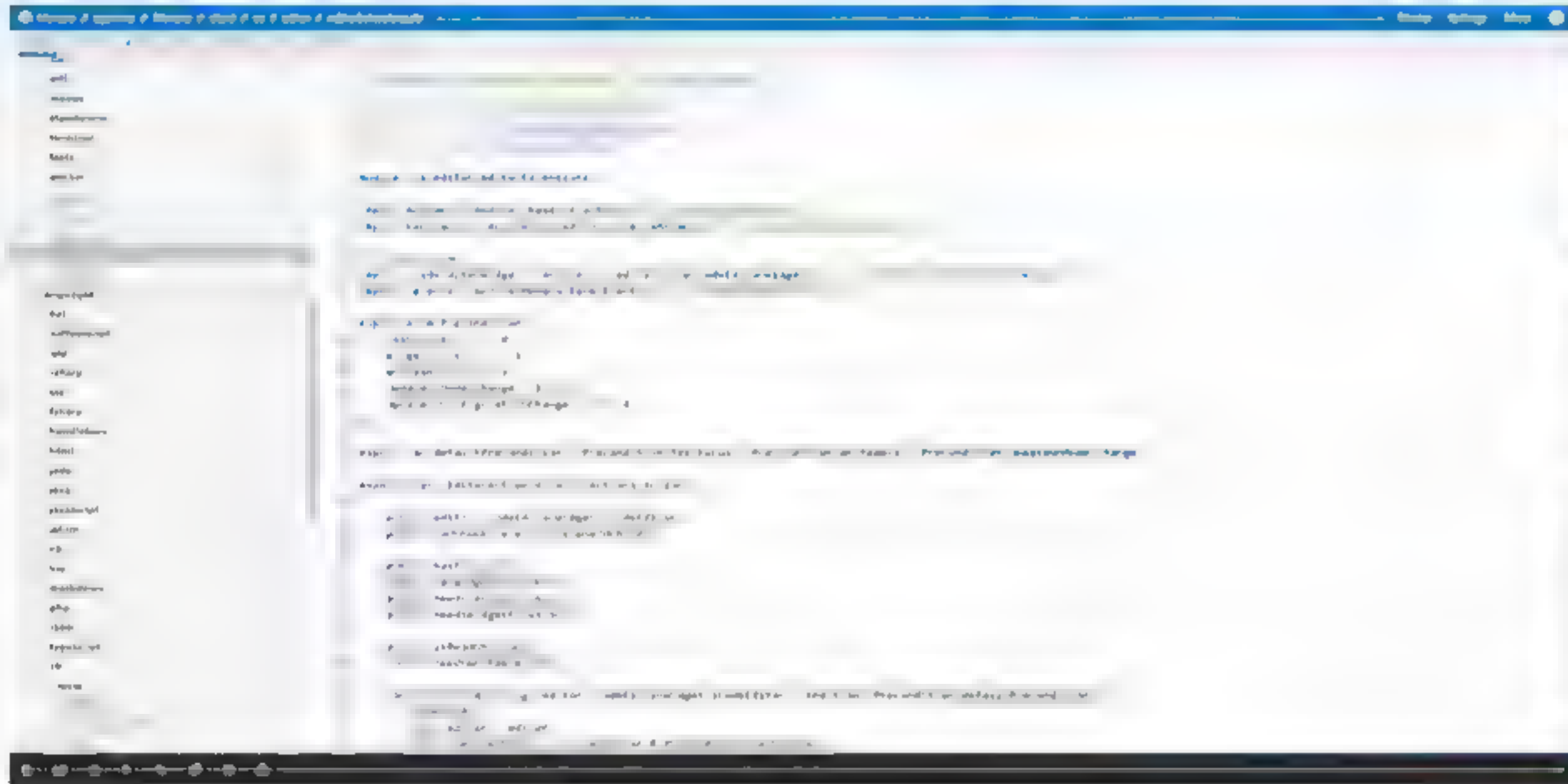
```
1 function insert(item, user, request) {  
2     if (item.userId !== user.userId) {  
3         request.respond(statusCodes.FORBIDDEN, 'You may only insert records with your userId.');4     } else {  
5         request.execute();  
6     }  
7 }
```

# Office 365 - Napa





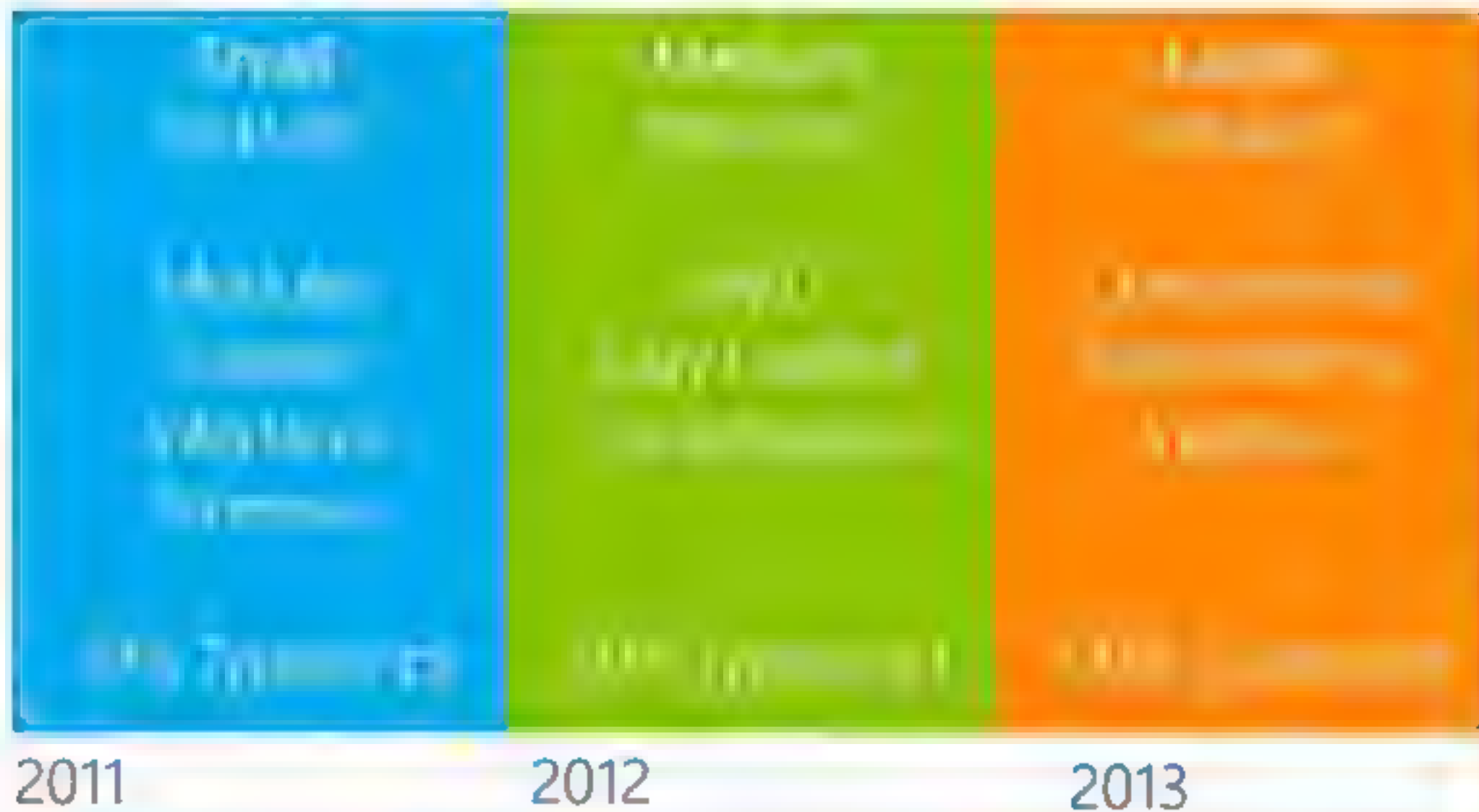
# Monaco @ Work - Dog Fooding



# Our Journey

patterns

TypeScript



# Inside

( , )

```
interface IRenderer {  
    getHeight(tree: ITree, element: any): number;  
    render(tree: ITree, element: any): void;  
}
```



# Interfaces at Work

Options: ICON

```
interface IOptions {  
    hideButtons?:bool;  
    okOnFocusLost?:bool;  
}
```

# Interfaces at Work

Fluent Interface

```
interface IMeasurementCallback {  
    (accessor: IMeasurementAccessor):void;  
}
```

# Interfaces at Work

Interface Definition

```
interface IEventManager {  
    [name: string]: EventListener[];  
}
```



# Interfaces at Work

Example: JQuery

```
interface JQuery {  
  addClass(classNames: string): JQuery;  
  addClass(func:(index: any, currentClass: any)=>JQuery);  
  attr(attributeName: string): string;  
  attr(attributeName: string, value: any): JQuery;  
  ....  
}
```

# Demo

Type definitions

# TypeScript Type Definitions

<https://github.com/borisyankov/DefinitelyTyped>

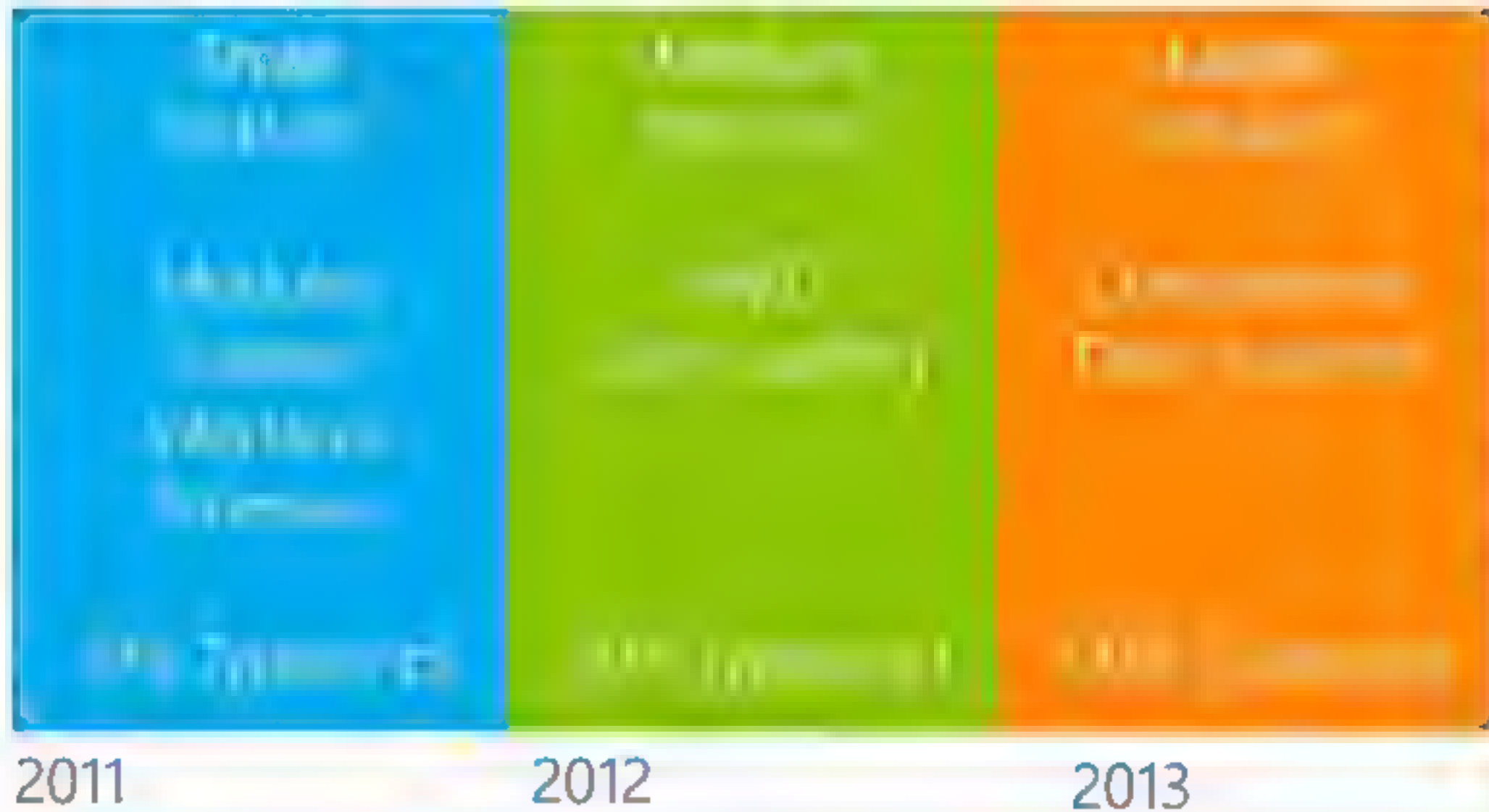
 <a href="#">#27</a>	2 months ago	amc harts have been added ( <a href="#">#27</a> )
 <a href="#">#100</a>	4 days ago	angular d.ts ngModule use of Object instead of < added tests ( <a href="#">#100</a> )
 <a href="#">#1</a>	2 months ago	Re async tests on test ( <a href="#">#1</a> )
 <a href="#">#1000</a>	4 months ago	Updated parameters for Updatable ( <a href="#">#1000</a> )
 <a href="#">#10000</a>	14 days ago	Definition of type for IE's <script> players implementation without ( <a href="#">#10000</a> )
 <a href="#">#10001</a>	3 months ago	conformed tests to readme md ( <a href="#">#10001</a> )
 <a href="#">#10002</a>	1 month ago	Added type definitions for lodash.map ( <a href="#">#10002</a> )
 <a href="#">#10003</a>	3 months ago	update reference paths ( <a href="#">#10003</a> )
 <a href="#">#10004</a>	11 hours ago	Added overload for modal so as to use bootstrap state ( <a href="#">#10004</a> )
 <a href="#">#10005</a>	4 month ago	updated box2d underscore sugar ( <a href="#">#10005</a> )
 <a href="#">#10006</a>	9 days ago	breeze updated 1.2.2 ( <a href="#">#10006</a> )
 <a href="#">#10007</a>	2 months ago	casperjs header was normalized ( <a href="#">#10007</a> )
 <a href="#">#10008</a>	2 days ago	Included Char Expect ( <a href="#">#10008</a> )
 <a href="#">#10009</a>	2 months ago	Update readme and metadata for Cheerio ( <a href="#">#10009</a> )
 <a href="#">#10010</a>	3 months ago	update reference paths ( <a href="#">#10010</a> )
 <a href="#">#10011</a>	3 months ago	Add type information for chrome's socket API ( <a href="#">#10011</a> )
 <a href="#">#10012</a>	3 months ago	Update reference paths ( <a href="#">#10012</a> )
 <a href="#">#10013</a>	2 month ago	Add commander.js definitions ( <a href="#">#10013</a> )
 <a href="#">#10014</a>	11 days ago	Add more D3 Translation methods ( <a href="#">#10014</a> )
<a href="#">#10015</a>	2 months ago	update demo definitions readme and metadata ( <a href="#">#10015</a> )



# Our Journey

patterns

TypeScript



# Growing Pains

## Managing scripts and their order

Manually edited lists of scripts

manifest.json

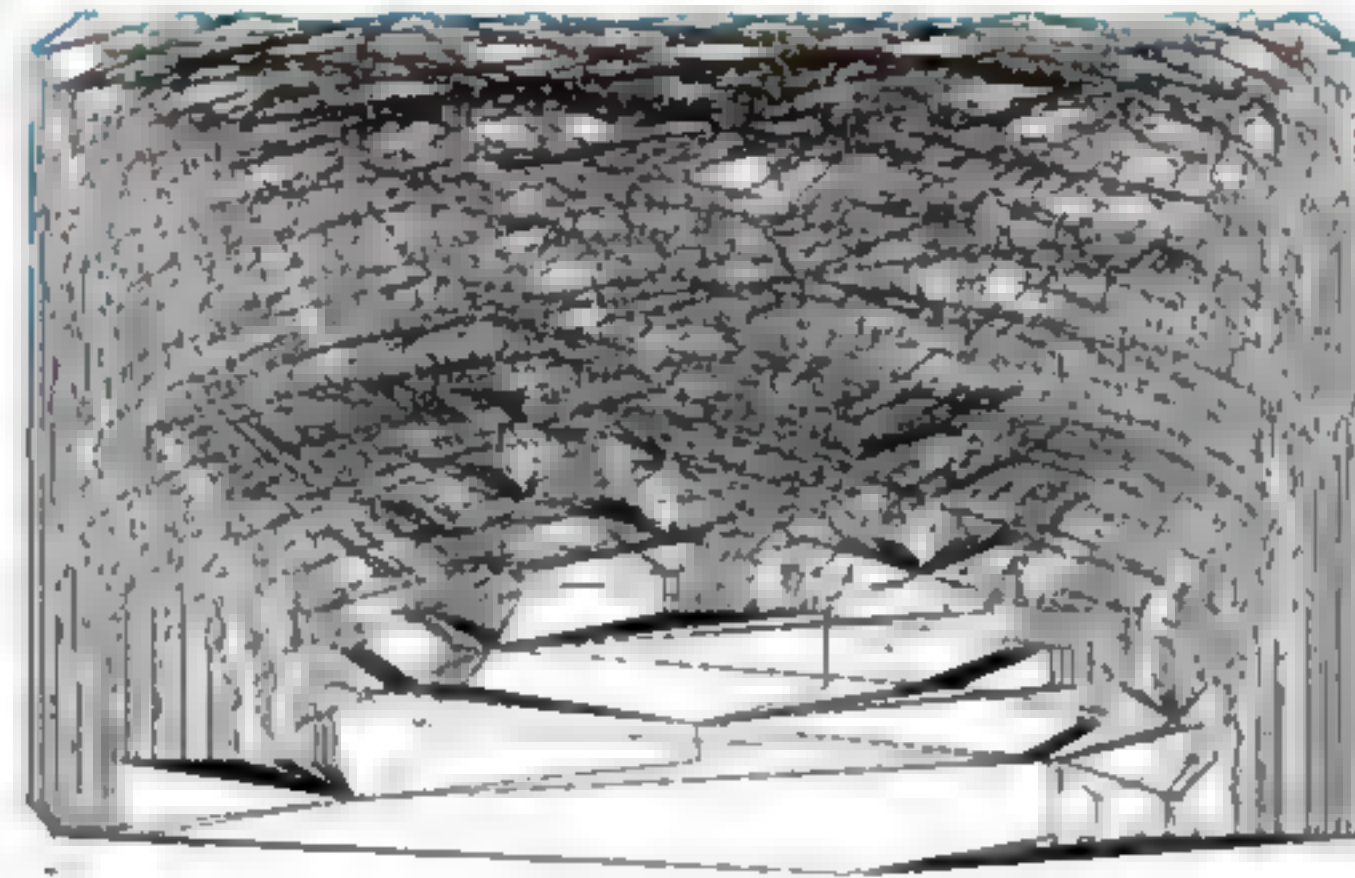
37

```
    "public/js/workbench.js",
    "public/js/constants.js",
    "public/js/platform.js",
    "public/js/events.js",
    "public/js/core/context.js",
    "public/js/core/storage.js",
    "public/js/core/history.js",
    "public/js/model/workspacemodel.js",
    "public/js/ui/safe.js",
    "public/js/ui/commands.js",
    "public/js/ui/notebook.js",
    "public/js/ui/layout.js",
    "public/js/ui/actions/action.js",
    "public/js/ui/menu.js",
    "public/js/ui/panels/view.js",
    "public/js/ui/panels/part.js",
    "public/js/ui/panels/sidebartab.js",
    "public/js/ui/panels/editor/baseditor.js",
    "public/js/ui/panels/editor/editorModel.js",
    "public/js/ui/panels/editor/editorInput.js",
    "public/js/ui/panels/editor/editorOptions.js",
    "public/js/ui/panels/editor/textEditor.js",
    "public/js/ui/panels/editor/stringEditorModel.js",
    "public/js/ui/panels/editor/stringEditorInput.js",
```

# Growing Pains: Dependencies...

It is easy to import an internal module...

It is easy to contribute to a module...



"our dependency graph was such a mess that each area had a dependency on just about every other area." –Nick

# AMD to the Rescue

`define(id?, dependencies?, factory)`

```
define([  
  'vs/base/lib/winjs', vs/editor/zoneWidget'],  
  function(WinJS, ZoneWidget) { ... }  
);
```



# TypeScript: External Modules

TypeScript supports code generation for different module systems

```
import widget= import('vs/base/widget');
```

```
import http= import('http');
```

```
tsc --module amd app.ts | define(..., function(...) {...}
```

```
tsc --module commonjs HttpServer.ts | var http= require("http");
```

# Before/After

## AMD in raw script

```
define(['../winjs.base', '../zoneWidget'],  
    function(WinJS, ZoneWidget) { ... }  
);
```

## TypeScript

```
import WinJS= import( vs/base/lib/winjs );  
import ZoneWidget = import( vs/editor/zoneWidget');
```

# AMD Applied – Self Contained Modules

Support à la carte consumption

Expressing CSS dependencies

CSS AMD loader plugin

TypeScript pragma

```
// <amd-dependency path="vs/css!./actionbar" />
```

# Growing Pains

## Start up time

## Eager loading of scripts and CSS

Slow startup: #requests, data transfer

Item	Material	Unit	Price	Quantity	Total	Remarks
1. Cement	OPC 42.5	50kg bag	1200	100	120000	
2. Sand	Medium Sand	cum	1500	10	150000	
3. Aggregate	20mm	cum	2500	5	1250000	
4. Labour	Unskilled	man/day	200	1000	200000	
5. Formwork	15mm Ply	sqm	100	100	10000	
6. Transportation	Truck	trip	500	10	5000	
7. Water	Piped Water	litre	10	10000	100000	
8. Electricity	Power	kwh	10	1000	10000	
9. Contingency					100000	
10. Profit					100000	
<b>Total</b>					<b>2500000</b>	



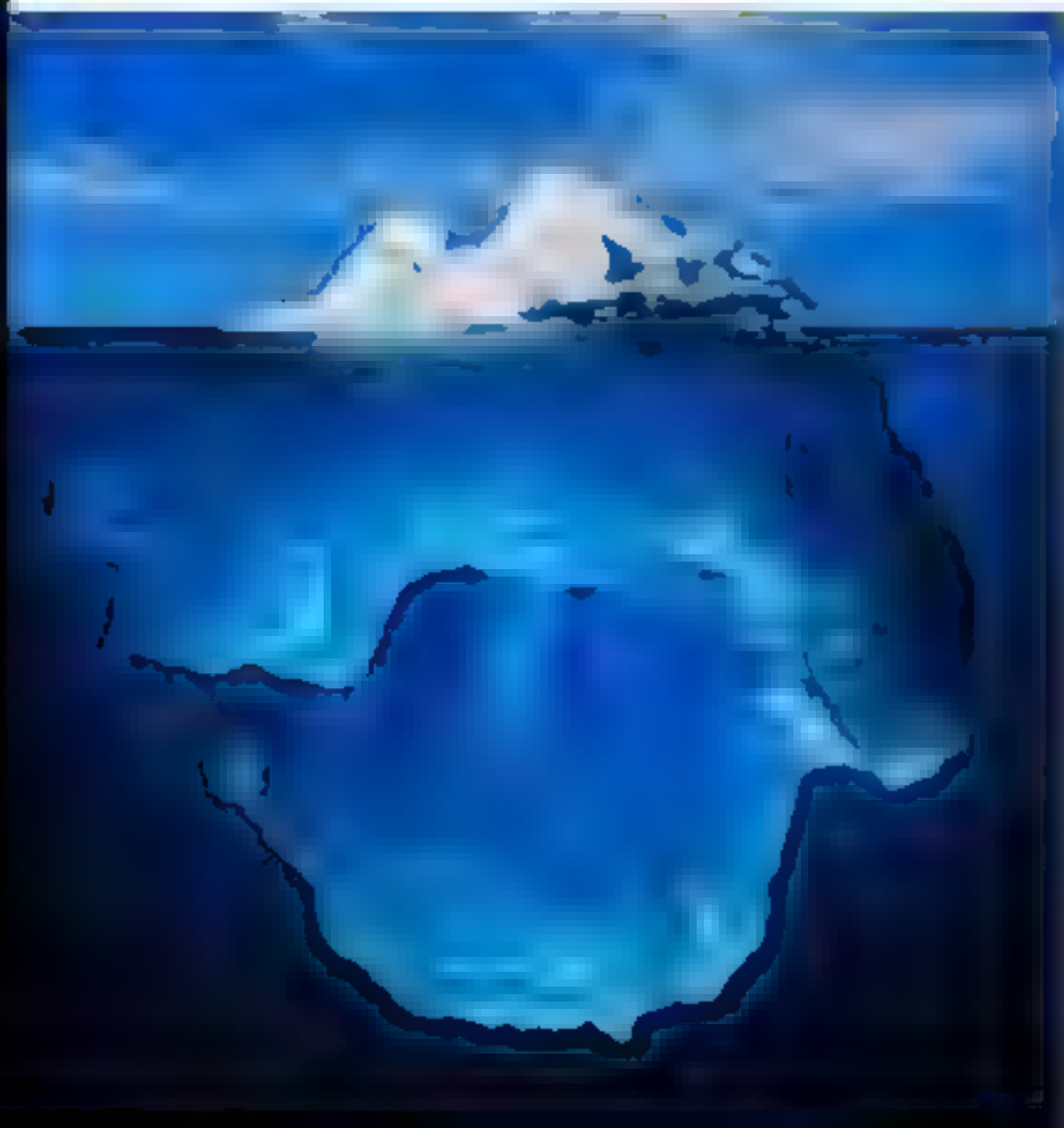
# Lazy Loading Contributions

`csharp.contribution.ts`

```
modeRegistry.registerMode(  
    ['text/x-csharp'],  
    new Platform.Descriptor(  
        'vs/languages/csharp/csharp',  
        'CSMode')  
    );
```

`csharp.ts`

```
export class CSMode extends  
modesExtensions.AbstractMode {  
    constructor() {  
        super('vs.languages.csharp');  
    }  
    // lots of code ....  
}
```



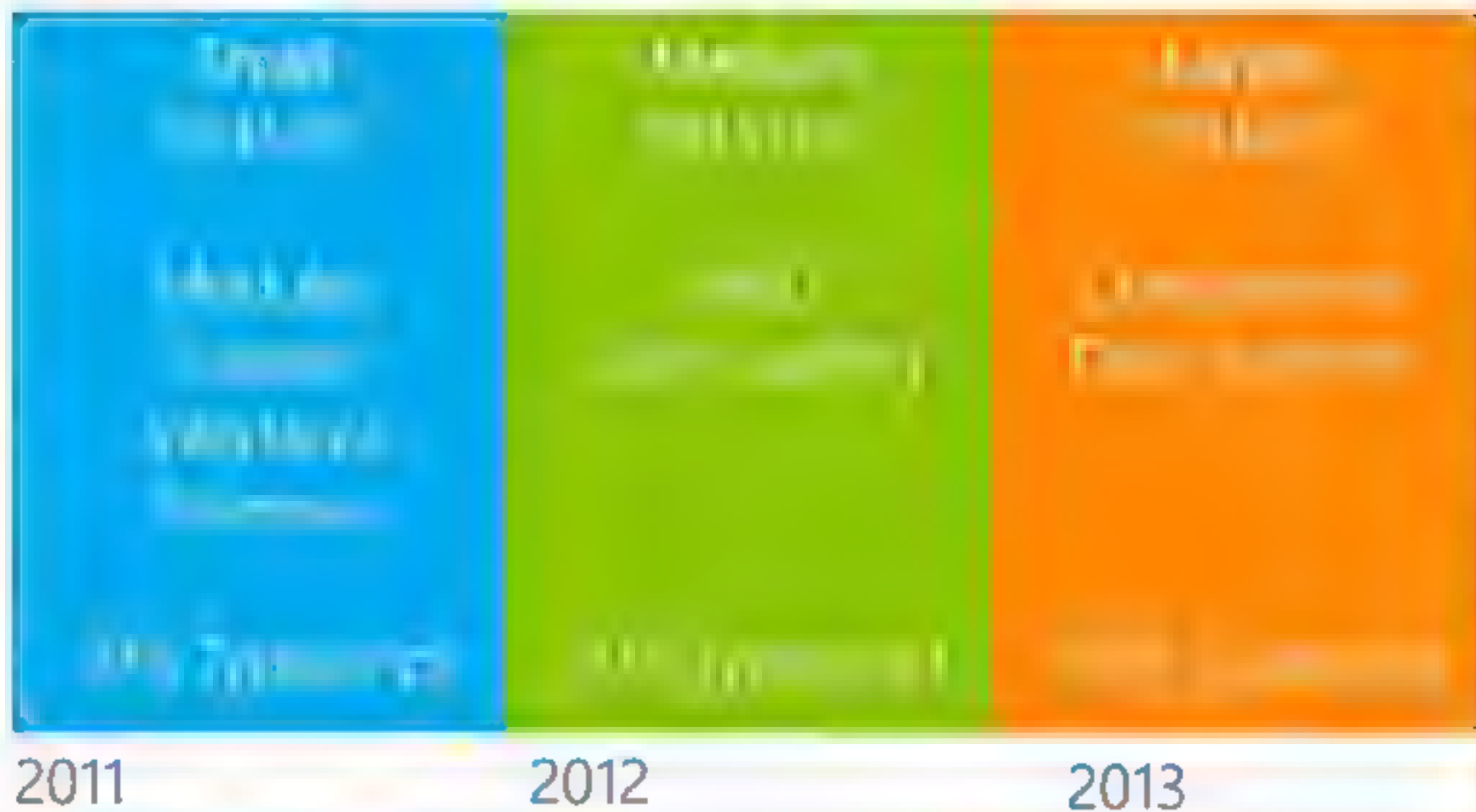
# After the AMD Migration Impressions

"It feels like [travis](#) showered. Self contained modules, no more cycles, no more globals, clean file system structure" -Alex

# Our Journey

patterns

TypeScript



# 100% TypeScript

"Writing JavaScript code in a large project is like carving code in stone" –Alex

Migration is code clean-up and real work

Velocity around 300 LOCs per hour

"As I did conversions, I began typing various object literals I was passing around as interfaces. Soon enough, I realized how inconsistent I was, the same data was flowing around in at least 3 different formats. This is because of the easiness through which you can create literals in JavaScript .... Need some placeholder for data?... Just create a new literal object." --Alex



# Even More Growing Pains

“Writing JavaScript code in a large project is like carving code in stone” –Alex

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Hard coded dependencies on context

Decreased testability

Difficulty to reuse components in different contexts

# Componentization

Tooling slow down, translation time

Tools need to digest lots of source to infer types

Reuse TypeScript code as 'binary' JS  
components with a declarations file

Example using TypeScript language services as a component

Compiler and language services > 30kLOC of TypeScript

```
tsc --declarations --out typescriptservices.js typescript.ts
```

# Dependency Injection

Dependencies on *services* provided by container

Decreased testability

Need for configuration flexibility

Services (20)

Selection, Progress, Logging, History, Events, Configuration, Telemetry, Requests, ...



# Dependency Injection

```
export interface IProgressService {  
    show(total?:number, delay?:number):IProgressRunner;  
}
```

```
export interface IProgressServiceConsumer {  
    injectProgressService(service:IProgressService):void;  
}
```

```
export class Explorer implements Services.IProgressServiceConsumer {  
    private progressService:Services.IProgressService;  
    public injectProgressService(service:Services.IProgressService) {  
        this.progressService = service;  
    }  
    //....  
}
```



# TS Retrospective

We were on the **bleeding edge**...

...but we expected it and had plenty of band aid

We would **do it again**, the benefits outweigh the pains

Code readability, refactoring agility, tooling, fun

**We would start with TypeScript (and AMD) from the beginning**

<http://www.typescriptlang.org/>